REMARKS

This amendment is submitted in response to an Office Action mailed August 29, 2002. Applicant respectfully requests reconsideration of the subject application as amended herein.

Claims 1-31 remain in the present application.

Dependent claims 7-15 and 22-30 were allowable.

As part of this amendment, the disclosure has been amended to correct previously undetected informalities. No new matter has been entered.

The August 29, 2002 Office Action objected to the Abstract. As part of this amendment, the Abstract has been amended. No new matter has been entered. Applicant respectfully submits that the amended abstract overcomes the objection.

The August 29, 2002 Office Action objected to claims 1-31. Applicant has amended claims 1, 7, 8, 12-15, 16, 22, 23, 27-30, and 31 merely to overcome the objections to claims 1-31.

Specifically, claims 7, 8, 22, and 23 have been amended to convert the claims to independent form and to place the claims, and their dependent claims 9-15 and 24-30, in condition for allowance. The amendments to claims 7, 8, 22, and 23 have not been made to overcome a statutory rejection and therefore do not limit the equivalence of any claim element under the Doctrine of Equivalence.

The preambles of the independent claims 1, 7, 8, 16, 22, 23, and 31 have been amended to clearly define the intent of the claims merely by restating elements as originally claimed. The amendments to claims 1, 7, 8, 16, 22, 23,

and 31 have not been made to overcome a statutory rejection and therefore do not limit the equivalence of any claim element under the Doctrine of Equivalence.

The preambles of dependent claims 12-15 and 27-30 have been amended merely to use the singular form of the verb "comprise" to overcome the objection. The amendments to claims 12-15 and 27-30 have not been made to overcome a statutory rejection and therefore do not limit the equivalence of any claim element under the Doctrine of Equivalence.

In the August 29, 2002 Office Action, claims 1-6, 16-21, and 31 were rejected under 35 U.S.C. § 103 as being unpatentable over U.S. Patent No. 6,182,268 issued to Kenneth S. McElvain (hereinafter "McElvain"). As discussed below, Applicant respectfully submits that claims 1-6, 16-21, and 31 are not obvious over McElvain.

Taking claim 1 as an example, claim 1 recites:

> A method for converting a data structure from a specific format in a hardware description language (HDL) to generic HDL elements, the method comprising:
>
> receiving the data structure representing a behavior of a circuit element, said circuit element being sequential and said data structure being defined using the specific format;
>
> generating a conversion matrix from the data structure, said conversion matrix to represent the behavior of the circuit element in a generic format; and
>
> determining a generic HDL register and a plurality of generic HDL input logic for the generic HDL register to replicate the behavior represented by the data structure based on the conversion matrix.

Claim 1 converts an HDL data structure from a specific format into generic HDL elements.

McElvain, in contrast, is directed to an entirely different goal. Specifically, McElvain is directed to identifying and optimizing state machines in HDL circuit designs (McElvain; col. 2, lines 24-32).

In McElvain, an HDL circuit design is compiled into a register transfer level (RTL) netlist or gate-level description (McElvain; col. 4, lines 39-46). State machines are then identified in the compiled HDL (McElvain; col. 6, lines 25-37). Once all possible states of a state machine have been determined, a symbolic representation of the state machine is created (McElvain; col. 11, lines 47-49). From the symbolic representation, it may be possible to optimize the state machine (McElvain; col. 12, lines 29-35). Finally, the symbolic representation can be used to recompile the state machine into compiled HDL (McElvain; col. 12, lines 35-41).

McElvain mentions that the circuit design may be constrained to a particular library for a target architecture (McElvain; col. 4, lines 16-24; col. 12, lines 42-48). When HDL is compiled, or the symbolic representation of a state machine is recompiled, the compiled HDL comprises components from the target architecture's library.

McElvain, however, does not suggest or disclose an HDL data structure defined using "a specific format," as claimed in claim 1. Rather, McElvain describes a circuit description compiled from HDL so that the compiled circuit description uses a particular library. The HDL itself is not clearly constrained in any way. Only the result from compiling the HDL is constrained to a library. In

other words, McElvain describes converting generic HDL to a specific form of compiled HDL.

Even assuming purely for the sake of argument that McElvain's target library could include specific formats of HDL data structures, McElvain does not suggest, disclose, or enable "converting" an HDL data structure from "a specific format to generic HDL elements," as claimed in claim 1. Moreover, McElvain suggests no motivation for performing such a conversion.

Thus, for at least the reasons discussed above, Applicant respectfully submits that claim 1 is not obvious in light of McElvain.

Applicant submits that the reasoning presented above with respect to claim 1 similarly applies to claims 16 and 31. Thus, for at least the reasons discussed above, Applicant respectfully submits that claims 16 and 31 are likewise not obvious in light of McElvain.

Given that claims 2-6 depend from claim 1, and claims 17-21 depend from claim 16, Applicant respectfully submits that claims 2-6 and 17-21 are likewise not obvious in light of McElvain for at least the reasons discussed above.

In conclusion, Applicant respectfully submits that claims 1-31 are now in a condition for allowance, and Applicant respectfully requests allowance of such claims.

Please charge any shortages and credit any overages to our Deposit Account No. 500393.

Respectfully submitted,

SCHWABE, WILLIAMSON & WYATT, PC

Date: 12/30, 2002

Robert A. Diehl
Reg. No. 40,992

10260 SW Greenburg Road, Suite 820
Portland, Oregon 97223
Phone: (503) 595-2800
FAX: (503) 595-2804

MARKED VERSION OF AMENDMENTS TO SHOW CHANGES MADE

IN THE ABSTRACT

Please amend the abstract as follows:

An HDL conversion process receives a data structure representing the behavior of a circuit element. The circuit element is sequential and the data structure is ~~define~~ defined in a hardware description language (HDL) using a specific format. The conversion process generates a conversion matrix from the data structure. The conversion matrix represents the behavior of the circuit element in a generic format. The conversion process then determines a generic HDL register and a plurality of generic HDL input logic for the generic HDL register to replicate the behavior represented by the data structure based on the conversion matrix.

IN THE SPECIFICATION

Please amend the specification as follows:

The paragraph at Page 12, Line 12 to Page 13, Line 2:

For instance, referring back to Figure 1, the UDP table defines N = 3 inputs and one output. Expanding the UDP out to a conversion matrix will result in a 3^4 by 3^4 (81 by 81) conversion matrix. For example, line 1 of the UDP table can be expanded out to three lines in the UDP table for the three possible values

represented by "?" in the Q column. The state of line 1 for D, CLK, SET, and Q can be written as a set of four bits. For instance, the current state for line 1 of D = 1, CLK = 0, SET = 0, and Q = ? could be written as 100?. Expanding the question mark out, the three lines would be 1001, 1000, and 100X. Each line is an "edge" line in that an edge transition is defined in the line for the CLK input to transition from 0 to 1, called a rising or positive edge. In which case, the next states for each respective line in the table would be 1101. Since each line in the table defines ~~on~~ one particular state transition, the conversion matrix has three entries, one at the intersection of current state row 1001 and next state column 1101, row 1000 and column 1101, and row 100X and column 1101.

The paragraph at Page 24, Lines 20-23:

At block 1110, the process receives a data structure representing the behavior of a sequential circuit element written in a specific HDL format. ~~At~~ As discussed above, the data structure may have come, for instance, from a library of HDL elements from a particular integrated circuit fabricator based on a particular fabrication technology.

IN THE CLAIMS

Please amend the claims as follows:

1. (Once Amended) A method for converting a data structure from a specific format in a hardware description language (HDL) to generic HDL elements, the method comprising:

receiving ~~a~~ the data structure representing a behavior of a circuit element, said circuit element being sequential and said data structure being defined ~~in a hardware description language (HDL)~~ using ~~a~~ the specific format;

generating a conversion matrix from the data structure, said conversion matrix to represent the behavior of the circuit element in a generic format; and

determining a generic HDL register and a plurality of generic HDL input logic for the generic HDL register to replicate the behavior represented by the data structure based on the conversion matrix.

7. (Once Amended) A method for converting a data structure from a specific format in a hardware description language (HDL) to generic HDL elements, the method comprising:

receiving the data structure representing a behavior of a circuit element, said circuit element being sequential and said data structure being defined using the specific format;

generating a conversion matrix from the data structure, said conversion matrix to represent the behavior of the circuit element in a generic format; and

determining a generic HDL register and a plurality of generic HDL input logic for the generic HDL register to replicate the behavior represented by the data structure based on the conversion matrix;

wherein generating the conversion matrix comprises:

identifying input signals and an output signal of the circuit element from the data structure;

evaluating state transitions for the circuit element by evaluating the data structure for a next output signal for each transition of the input signals and for each current output signal; and

populating entries of the conversion matrix for each state transition; and

~~The method of claim 6~~ wherein states for individual signals comprise 0, 1, and X.

8. A method for converting a data structure from a specific format in a hardware description language (HDL) to generic HDL elements, the method comprising:

receiving the data structure representing a behavior of a circuit element, said circuit element being sequential and said data structure being defined using the specific format;

generating a conversion matrix from the data structure, said conversion matrix to represent the behavior of the circuit element in a generic format;

determining a generic HDL register and a plurality of generic HDL input logic for the generic HDL register to replicate the behavior represented by the data structure based on the conversion matrix;

~~The method of claim 1~~ wherein the conversion matrix is organized into a plurality of current states of the circuit element and corresponding next states of the circuit element, and wherein determining the generic HDL register and the plurality of generic HDL input logic comprises:

classifying each next state of the conversion matrix into one of a plurality of sets of states based on predefined criteria;

selecting either an edge sensitive HDL primitive or a level sensitive HDL primitive for the generic HDL register based on selected ones of the plurality of sets of states;

selecting a particular set of functions based on the generic HDL register selected, each function of the particular set of functions corresponding to an input to the generic HDL register; and

evaluating the particular set of functions to determine the plurality of generic HDL input logic.

12. (Once Amended) The method of claim 9 wherein the particular set of functions for the edge sensitive HDL primitive ~~comprise~~ comprises:

Fset = L01 with do not cares for a union of L11, LUR01, and LUR11;

Frst = L10 with do not cares for a union of L00, LUR10, and LUR00;

Fclen = a single common input identified in sets E01 and E10;

Ftmpa = selected next states from a union of E01, E11, and L11;

Ftmpb = Ftmpa with do not cares for selected next states from a union of EUR01, EUR11, and LUR11;

Fd = Ftmpb with do not cares for a union of Fset and Frst, wherein the single common input identified for Fclen is ignored, and wherein the selected next states are edge states having a same edge transition as the single common input identified for Fclen

13. (Once Amended) The method of claim 9 wherein the particular set of functions for the level sensitive HDL primitive comprises ~~comprise~~:

Fset = L01 with do not cares for L11;

Frst = L10 with do not cares for L00;

Fclen = 0; and

Fd = 0.

14. (Once Amended) The method of claim 9 wherein the particular set of functions for the level sensitive HDL primitive comprises ~~comprise~~:

Fset = 0;

Frst = 0;

Fclen = (L01 with do not cares for L11) union with (L10 with do not cares for L00); and

Fd = L01 with do not cares for L11.

15. (Once Amended) The method of claim 9 wherein the particular set of functions for the level sensitive HDL primitive comprises ~~comprise~~:

Fset = a single input identified from the connectivity matrix for which there is no current state for which the output Q+ of the corresponding next state is one;

Frst = a single input identified from the connectivity matrix for which there is no current state for which the output Q+ of the corresponding next state is zero;

Ftmp1 = L01 with do not cares for a union of L11, LUR01, and LUR11;

Ftmp2 = L10 with do not cares for a union of L00, LUR10, and LUR00;

Fclen = a union of Ftmp1 and Ftmp2 with do not cares for a union of Fset and Frst; and

Fd = Ftmp1 with do not cares for an inverted Fclen.

16. (Once Amended) A machine readable medium having stored thereon machine executable instructions to implement a method for converting a data structure from a specific format in a hardware description language (HDL) to generic HDL elements, the method comprising:

receiving ~~a~~ the data structure representing a behavior of a circuit element, said circuit element being sequential and said data structure being defined ~~in a hardware description language (HDL)~~ using ~~a~~ the specific format;

generating a conversion matrix from the data structure, said conversion matrix to represent the behavior of the circuit element in a generic format; and

determining a generic HDL register and a plurality of generic HDL input logic for the generic HDL register to replicate the behavior represented by the data structure based on the conversion matrix.

22. (Once Amended) A machine readable medium having stored thereon machine executable instructions to implement a method for converting a data structure from a specific format in a hardware description language (HDL) to generic HDL elements, the method comprising:

receiving the data structure representing a behavior of a circuit element, said circuit element being sequential and said data structure being defined using the specific format;

generating a conversion matrix from the data structure, said conversion matrix to represent the behavior of the circuit element in a generic format;

determining a generic HDL register and a plurality of generic HDL input logic for the generic HDL register to replicate the behavior represented by the data structure based on the conversion matrix;

wherein generating the conversion matrix comprises:

identifying input signals and an output signal of the circuit element from the data structure;

evaluating state transitions for the circuit element by evaluating the data structure for a next output signal for each transition of the input signals and for each current output signal; and

populating entries of the conversion matrix for each state transition; and

~~The machine readable medium of claim 21~~ wherein states for individual signals comprise 0, 1, and X.

23. (Once Amended) A machine readable medium having stored thereon machine executable instructions to implement a method for converting a data structure from a specific format in a hardware description language (HDL) to generic HDL elements, the method comprising:

receiving the data structure representing a behavior of a circuit element, said circuit element being sequential and said data structure being defined using the specific format;

generating a conversion matrix from the data structure, said conversion matrix to represent the behavior of the circuit element in a generic format;

determining a generic HDL register and a plurality of generic HDL input logic for the generic HDL register to replicate the behavior represented by the data structure based on the conversion matrix;

~~The machine readable medium of claim 16~~ wherein the conversion matrix is organized into a plurality of current states of the circuit element and corresponding next states of the circuit element, and wherein determining the generic HDL register and the plurality of generic HDL input logic comprises:

classifying each next state of the conversion matrix into one of a plurality of sets of states based on predefined criteria;

_____ selecting either an edge sensitive HDL primitive or a level sensitive HDL primitive for the generic HDL register based on selected ones of the plurality of sets of states;

_____ selecting a particular set of functions based on the generic HDL register selected, each function of the particular set of functions corresponding to an input to the generic HDL register; and

_____ evaluating the particular set of functions to determine the plurality of generic HDL input logic.

27. (Once Amended) The machine readable medium of claim 24 wherein the particular set of functions for the edge sensitive HDL primitive comprises ~~comprise~~:

Fset = L01 with do not cares for a union of L11, LUR01, and LUR11;

Frst = L10 with do not cares for a union of L00, LUR10, and LUR00;

Fclen = a single common input identified in sets E01 and E10;

Ftmpa = selected next states from a union of E01, E11, and L11;

Ftmpb = Ftmpa with do not cares for selected next states from a union of EUR01, EUR11, and LUR11;

Fd = Ftmpb with do not cares for a union of Fset and Frst, wherein the single common input identified for Fclen is ignored, and wherein the selected next states are edge states having a same edge transition as the single common input identified for Fclen

28. (Once Amended) The machine readable medium of claim 24 wherein the particular set of functions for the level sensitive HDL primitive comprises ~~comprise~~:

Fset = L01 with do not cares for L11;

Frst = L10 with do not cares for L00;

Fclen = 0; and

Fd = 0.

29. (Once Amended) The machine readable medium of claim 24 wherein the particular set of functions for the level sensitive HDL primitive comprises ~~comprise~~:

Fset = 0;

Frst = 0;

Fclen = (L01 with do not cares for L11) union with (L10 with do not cares for L00); and

Fd = L01 with do not cares for L11.

30. (Once Amended) The machine readable medium of claim 16 wherein the particular set of functions for the level sensitive HDL primitive comprises ~~comprise~~:

Fset = a single input identified from the connectivity matrix for which there is no current state for which the output Q+ of the corresponding next state is one;

Frst = a single input identified from the connectivity matrix for which there is no current state for which the output Q+ of the corresponding next state is zero;

Ftmp1 = L01 with do not cares for a union of L11, LUR01, and LUR11;

Ftmp2 = L10 with do not cares for a union of L00, LUR10, and LUR00;

Fclen = a union of Ftmp1 and Ftmp2 with do not cares for a union of Fset and Frst; and

Fd = Ftmp1 with do not cares for an inverted Fclen.

31. (Once Amended) An apparatus for converting a data structure from a specific format in a hardware description language (HDL) to generic HDL elements, the apparatus comprising:

a processor; and

a machine readable storage medium storing thereon machine executable instructions, the processor to execute the machine executable instructions to

receive ~~a~~ the data structure representing a behavior of a circuit element, said circuit element being sequential and said data structure being defined ~~in a hardware description language (HDL)~~ using ~~a~~ the specific format;

generate a conversion matrix from the data structure, said conversion matrix to represent the behavior of the circuit element in a generic format; and

determine a generic HDL register and a plurality of generic HDL input logic for the generic HDL register to replicate the behavior represented by the data structure based on the conversion matrix.